REMARKS

Applicants respectfully request reconsideration of this application, as amended.

Applicants respectfully submit based on the previously submitted remarks that obviousness-type double patenting does not exist between the subject application and U.S. Patent 6,374,261. However, it is respectfully requested this rejection be held in abeyance until the indication of allowable subject matter.

Regarding the rejection of claims 9 and 10 under 35 U.S.C §112, Applicants respectfully submit the claims are fully compliant with 35 U.S.C §112. In particular, Independent Claim 8 recites two steps, steps (i) and (ii), which are alternative features that can be used in combination. With dependent claims 9 and 10 specifically requiring substep (i) and substep (ii) be performed, respectively, by definition, since a substep is required by the dependent claim, it further limits independent claim 8 in that the step was optional in the independent claim.

Regarding the rejection of claims 12, 23-27 under 35 U.S.C §112, first paragraph, support for the claimed features can at least be found on pages 12, last paragraph, and 20, first paragraph. Specifically, on page 12 it is stated that differing input structures can be embedded within one another and page 20 states that for an exemplary embodiment no external tag is supplied to the parser with the input string.

Claim 26 has been amended in accordance with the Examiner's recommendation.

Withdrawal of the rejections under 35 U.S.C §112 is respectfully requested.

Independent claim 1 recites a plurality of parsers operable to parse an input stream, each parser corresponding to a unique input structure;

a parser selection agent operable to receive the input stream and select a subset of the plurality of parsers to parse the input stream, wherein the input stream comprises a plurality of differing input structures and wherein the selected subset of parsers produce multiple parser outputs corresponding to the plurality of differing input structures; and

an encoding agent operable to convert the multiple parser outputs to a common grammar.

Johnson at least fails to teach or suggest the use of a meta-parser to select an arbitrary input stream corresponding to a plurality of grammars to produce respective abstract syntax trees followed by an encoder converting the abstract syntax trees to a common grammar.

The Office Action asserts that this feature is taught at paragraphs [0037] to [0044].

> [0037] When a message is to be processed by a message flow, it is first necessary to decode the message bit stream using one or more message parsers 50 to enable the various components of the message to be understood. Message type and format information for messages predefined in the message repositories is typically included in the messages' headers, and so a parser can recognise the message structure and format when the message is received. Messaging products such as IBM's MQSeries Integrator product are known to include a number of message parsers 50 which can parse known message structures including specific headers. However, with increasing systems and network integration, it is a requirement for these solutions to be extensible to add new parsers for new message structures and new data formats within a message. A particular extensible solution which is implementable within computer program products such as IBM's MQSeries Integrator products, and which satisfies this message broker parsing requirement will now be described in detail.
>
> [0038] FIG. 2A shows an example message structure of a message as taken from an input message queue. This comprises a message descriptor 100 (MD), one or more additional message headers 110 (in this case RFH2), and a message body 120. Normally the message data within the body part of the message has a single format, and so a single parser will handle the parsing of the entire data contents. Nevertheless, as the bit stream of the message is analysed to identify the separate MD, headers and body components, each of these components will be assigned to an appropriate parser. Additionally, the message as modelled by parsing will include a Properties component which is generated from data provided by the input node of the message flow (see next paragraph--this is possible if only messages having certain properties are input to this input node) and/or from data extracted from the message headers. The Properties component will be handled by an appropriate parser which is typically different from the parsers handling the MD, headers, and body. If the message body contains multiple nested formats, then multiple parsers will also be required to parse the body. The implementation of parser selection and invocation for handling nested message formats will be described below.
>
> [0039] Within a simple example message flow of a message broker comprising an input node, a processing node and an output node, the processing node may be a Compute node which transforms a message from one format to another, so that sending and receiving applications can communicate with each other using their own formats. Such a simple message flow is shown in FIG. 3. The input node requires the message to be parsed to understand its structure before the Compute node can perform its format transformation.
>
> [0040] On receipt of a message, the message broker 30 must pass the message bit stream (an array of bytes which make up the message) to a message parser 50. Consider the previous example of an incoming message which comprises:

[0041] A message descriptor (MD)

[0042] An RFH2 header

[0043] An XML data portion

[0044] A parser selector 80 of the broker 30 sends the MD component to an MD parser 50, which accesses a structure template for MDs which is stored in the message repository 60 and applies this to the received MD to model it as a sequence of ordered name-value pairs. The MD parser then reads one or more predefined message fields identifying the next message component's type and/or its format, compares this information with a list of component types/formats and selects and invokes another parser 50 which has a predefined responsibility for handling parsing of components having this type and/or format. The MD parser also specifies what portion of the bit stream it has consumed to indicate where a next selected parser 50 should begin. If the next component is the RFH2 header, then this component is given to a specific RFH2 parser. The RFH2 parser applies a stored RFH2 template to parse the header, modelling it as a sequence of name-value pairs, and then this RFH2 parser reads one or more fields relating to the next component. If the next component is the XML data portion, then the RFH2 parser invokes an XML parser in response to identifying that the component comprises XML. Since the templates stored for the MD and message headers determine what is the last field within their respective message component, the MD and RFH2 parsers can easily determine when to invoke the next parser.

In these passages, Johnson teaches using specific parsers to parse nested grammars and forming each grammar into a corresponding abstract syntax tree. However, as is abundantly clear, Johnson *does not* teach an encoding agent to convert the various different abstract syntax trees into a common grammar -Johnson does not even make mention of "encoding" anywhere in the application.

Independent claim 8 recites (a) receiving an input stream, the input stream comprising information defined by at least first and second input structures;

(b) providing at least a portion of the input stream to each of a plurality of parsers, the plurality of parsers corresponding to differing sets of grammars;

(c) receiving output from each of the plurality of parsers; and

(d) based on the outputs of the plurality of parsers, performing at least one of:

(i) selecting a first output from a first parser that corresponds to the first input structure and a second output from a second parser that corresponds to the second input structure; and

(ii) selecting a first parser corresponding to the first input structure to parse one or more first segments of the input stream and a second parser corresponding to the second input structure to parse one or more second segments of the input stream.

Johnson at least fails to teach providing a common part of the arbitrary input to multiple parsers and, based on the outputs of the parsers, selecting an appropriate parser.

The Office Action asserts that this feature is taught at paragraphs [0044], [0047] to [0050] and [0120]. These passages teach a chain-type approach to parser selection in which the parser selector 80 selects the initial parser. The initially selected parser parses the message body until a different grammar is encountered. That parser, and not the parser selector 80, then selects a next parser to handle the nested grammar, and so on.

At least based on the above distinctions, Johnson does not anticipate Claim 8.

Independent claims 23 recites receiving a stream of information, the stream being generated by one of a plurality of possible different computational sources, wherein each computational source generates a stream corresponding to a unique input structure and wherein each of a plurality of differently structured segments of the stream is free of an embedded tag indicating a corresponding computational source and/or input structure for the respective segment;

comparing at least a portion of the stream with a set of tokens to provide a subset of tokens identified in the at least a portion of the stream;

heuristically identifying, from among at least one of a plurality of possible input structures and a plurality of possible computational sources, at least one of an input structure corresponding to the at least a portion of the stream and a computational source for the at least a portion of the stream; and

parsing the stream based on the identified at least one of an input structure and computational source.

Independent Claim 28 recites an input operable to receive a stream of information, the stream being generated by one of a plurality of possible different computational sources, wherein each computational source generates a stream corresponding to a unique input structure; and

a parser operable to (a) compare at least a portion of the stream with a set of tokens to provide a subset of tokens identified in the at least a portion of the stream;

(b) heuristically identify, from among at least one of a plurality of possible input structures and a plurality of possible computational sources, at least one of an input structure corresponding to the at least a portion of the stream and a computational source for the at least a portion of the stream; and (c) parse the stream based on the identified at least one of an input structure and computational source, wherein the parser is not provided with an input structure identifier, other than the corresponding input structure itself, either in or external to the at least a portion of the input stream to identify or assist in the identification of the at least one of the respective input structure corresponding to the at least a portion of the stream and a computational source for the at least a portion of the stream.

Johnson at least fails to teach how to parse an input stream being free of an embedded tag indicating a corresponding computational source and/or input structure and comparing a selected portion of the input stream with multiple different sets of tokens corresponding to differing grammars to provide one or more subsets of tokens identified in the input stream and, based on the identified subsets of tokens, selecting a set of tokens to be used in parsing the remainder of the input stream.

While Johnson teaches that the message type and format information in the header is critical to the parser recognizing the message structure and format, e.g., ¶[0037] and [0044], these passages clearly illustrate that the message structure, and *not a parser output*, is key to selecting an appropriate parser.

At least based on the above, there are clear patentable distinctions between the Independent Claims and the teachings of Johnson - specifically, numerous claimed features are neither taught, suggested, nor disclosed in Johnson.

The dependent claims provide additional reasons for allowability.

By way of example, Johnson does not teach or suggest determining and assigning to a selected input stream segment a set of flags corresponding to a set of values depending on the presence or absence of a syntactical and/or semantical relationship. The flags are used to identify at least one of an input structure and a computational source for the segment. (See claims 25 and 30.)

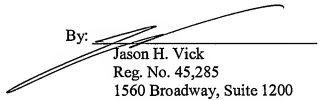
Dependent claims 16-19 are directed to recursive analysis of a parse tree as part of converting parsed output to a common grammar.

Based on the foregoing, Applicants believe that all pending claims are in condition for allowance and such disposition is respectfully requested. In the event that a telephone conversation would further prosecution and/or expedite allowance, the Examiner is invited to contact the undersigned.

Respectfully submitted,

SHERIDAN ROSS P.C.

Date: 27 March 2008

By: 

Jason H. Vick
Reg. No. 45,285
1560 Broadway, Suite 1200
Denver, Colorado 80202
Telephone: 303-863-9700